# **MILESTONE 4** -- SFT221 SCRUM Report and Reflection

All students are expected to attend the SCRUM meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP**: \_\_\_\_\_\_\_\_\_\_\_\_\_**1**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Song Nhat Nguyen | 4. Prince Ghumaan |
| 2. Nguyen Dang Khoa Huynh | 5. Benson Liu (Can not contact) |
| 3. Mohamed Mohamed | 6. |

## Milestone 4 Tasks

**Deliverables due 4 days after your lab day:**

* Finish implementing/coding the functions.
* Finish implementing/coding blackbox tests. Store in repo, executed, results in Jira (and on corresponding test documents, and debugged.
* A set of whitebox tests as test documents (in an Excel file) with test data for the functions you created. At least 4 sets of test data are required for each function. You must have test cases for at least 6 functions (including all your custom function). Stored in the repository.
* Whitebox tests implemented (in the C++ testing project), stored in repository, executed, results in Jira and on corresponding test documents, and debugged (at least 1 SET is required).
* Updated requirements traceability matrix stored in the repository.
* Completed hook file (for EACH team member) for test automation stored in the repository.
* Completed scrum report including reflection questions answered.

**Rubric:**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Implemented functions and main (well-designed, and documented) | 10% |
| Finish coding blackbox code (well-designed, written, and documented) | 5% |
| Whitebox test case document (well written, complete, good test data) | 10% |
| Whitebox test code (well designed and documented) | 20% |
| Updated requirements traceability matrix | 5% |
| Test execution (performed, results recorded, issues created) | 10% |
| Debugging (bugs fixed, documented, Jira updated) | 5% |
| Hook files | 10% |
| Git usage (used properly with good structure) | 5% |
| Jira usage (creates issues, tracks progress) | 10% |
| Scrum report & reflections | 10% |
| **Deadline** | 20% deduction for each day you are late |  |

**SCRUM Report**

**Summary of Tasks Completed or Delayed in the last week:**

Here you can list all of the tasks completed in the last week along with any tasks which could not be completed with a reason why they could not be completed.

|  |  |  |
| --- | --- | --- |
| **Member** | **Tasks Completed** | **Tasks Delayed/Blocked** |
| Song Nhat Nguyen | Function specs Implementation, Testing code(Black box, White Box) | **N/A** |
| Nguyen Dang Khoa Huynh | Support Testing Case, Fixing bugs when using Test Explorer, Scrum Report, Assign Task on Jira and upload files on GitHub repo. | **N/A** |
| Mohamed Mohamed | Reflection 1 & 2 | **N/A** |
| Prince Ghumaan | Reflection 3 & 4 | **N/A** |
|  |  |  |

For every task delayed or blocked, describe the reason for the delay or block, how it impacts the project and the proposed solution or workaround**.**

|  |  |
| --- | --- |
| **Delayed or Blocked Task** | **N/A** |
| **Reason for delay or block** | **N/A** |
| **Impact on Project** | **N/A** |
| **Solution or work-around** | **N/A** |
|  |  |
| **Delayed or Blocked Task** | **N/A** |
| **Reason for delay or block** | **N/A** |
| **Impact on Project** | **N/A** |
| **Solution or work-around** | **N/A** |

**Summary of Meeting:**

A summary of the main points discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Function specs implementation | Implementing function specs that was developed in Truck.h in last week. | Implementation finished and discussed in meeting. |
| SCRUM | SCRUM done | SCRUM Finished |
| Project Details | Discussed what would be needed to be created for submission | Clear understanding of task at hand |
| Jira | Task Schedule setup in Jira (Debug ticket) | Completed |
| GitHub | Git update to each branch (Debug tickets on Git project) | Completed |

**Summary of Decisions Made:**

This will include major architecture and design decisions, testing decisions, prioritization of tasks, dealing with problems encountered and other major outcomes from the meeting.

|  |  |
| --- | --- |
| Decision | Rationale |
| White Box testing | Need new testing implementation, executed, Unit Testing through Test Explorer from Visual Studio, export testing results and share with teammates. AddPackage, CanAddPackage, CapacityLeft, FindTruck has been debugged and fixed and pushed to master. Debugging record exist in Git project and jira. |
| Black Box Testing |  |
| Function Implementations | Implementation will follow truck.h that was developed last week as function specs |
|  |  |
|  |  |
|  |  |

**Tasks Attempted During Meeting:**

Each member is assumed to participate in the SCRUM meeting and contribute to the completion of the SCRUM report and reflections. Since the SCRUM meeting will not take more than 20-30 minutes, there is lots of time left to undertake some of the actual work tasks. In the table below, each member should list what they did to complete the SCRUM report, the reflections, and 1-4 other tasks they completed during the class period. If a task could not be completed, the student should indicate why this was not possible.

|  |  |  |  |
| --- | --- | --- | --- |
| Member | Task Attempted | Time Spent | Complete? |
| All | **Discussion on the analysis of debugging in black box testing conducted last week, together with an examination of white box implementation and execution.** | 1 hour | Yes |
| Song Nhat Nguyen and Nguyen Dang Khoa Huynh | **Discussion on analysis, implementation of the analysis function in the software, and further discourse.** | 1 hour | Yes |
| Nguyen Dang Khoa Huynh | **Scrum report** | 30 min | Yes |
| Song Nhat Nguyen and Nguyen Dang Khoa Huynh | **Discussion on hook automation** | 30 min | Yes |
| ALL | **Discussion for next week tasks** | 15 min | Yes |
|  |  |  |  |
|  |  |  |  |

**SCRUM Tasks Selected for Next Week**:

The tasks each member has selected to pursue for this class or the next week.

|  |  |
| --- | --- |
| Group Member | Task Description |
| Nguyen Dang Khoa Huynh | Jira control |
| All | Meeting on Monday class |
| All | SCRUM, Reflection |
| ALL | Acceptance Tests |
| ALL | Integration Tests |
| All | Debugging |
| All | Test Execution |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

This is where you should highlight the major accomplishments of the class.

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Function Implementation | The function was implemented in accordance with the specifications drafted last week. |
| Black Box Testing | The team completed the outline black box testing last week, despite it being intended for this week's requirements. We identified many defects in our testing codes, documented them in Jira Kanban, as well as in the Git project, and the issues have been rectified. |
| White box testing | White box testing codes were developed and run. |
| Hook implementation | Hook implementation was discussed, and screenshot was up to hookfiles on GitHub repo |
|  |  |

**Things That Went Well in This Meeting:**

Here you can highlight things which worked well. This indicates that the way you worked on these items is working and should be continued.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Success |
| Git | Beneficial for version control and monitoring modifications |
|  |  |
|  |  |
|  |  |

**Things That Did NOT go Well in This Meeting:**

This is where you can list things which did not go well in the class. You should analyze why this happened and suggest how you can improve it next time. This will lead to the goal of *continuous process improvement*.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Problem and How to do Better |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

1. After you run your blackbox and whitebox tests you are asked to record the results in both the original test document as well as in Jira. Explain why it is a good idea to record the results in both places.

It is a good idea to record the results in both the original test document and Jira because it is always good you have multiple files on record in case one gets corrupted. While Jira has easier access for collaboration, the notes you share with each other tend to be short form, that’s why writing on the document can allow for more detailed tests and can hold a lot more information revolving those tests. Having both Jira and test documents can be beneficial depending on who you have to show your results too, having two ways to absorb the information can help many people understand it

1. Why did we wait until the fourth milestone to write the whitebox tests?  
   We waited until the fourth milestone to write white box cases because that is when he have finalized the functions we have created. Waiting until blackbox testing has been done and have dealt with the bugs and errors fast. White box testing is also done when your code has reached its final stages to prevent the need of having to redo the testing afterwards incase you change code.

1. Pick one of the functions you created and list its name. For this function did you produce more blackbox or whitebox tests? Explain why your answer (more blackbox or more whitebox) happens for most functions.

In software testing, black box testing is conducted from the user's point of view, where the internal workings and code implementation details of the program are not known to the tester. The primary objective of black box testing is to ensure that the basic functionality of the program works as intended and meets the specified requirements. Testers focus on the program's inputs and expected outputs without considering how the code achieves those results.

By employing black box testing for the given function, mapping.c, we aimed to verify that its external behavior, as seen by the end-users, aligns with the expected functionality outlined in the milestone specification. This approach allows us to assess whether the function fulfills its intended purpose without delving into the intricacies of its internal implementation.

On the other hand, white box testing, as applied to the function finder.c, leverages knowledge of the internal code and logic. It enables us to thoroughly examine and validate the function's internal pathways, ensuring that the code is executing correctly and efficiently. This level of testing is especially valuable when dealing with custom-made data structures and newly implemented functions, as it helps us identify potential bugs, corner cases, and performance optimizations by scrutinizing the code at a granular level.

In software testing, black box testing is performed from the user's perspective, without knowledge of the program's internal mechanisms or code implementation specifics. The main aim of black box testing is to verify that the fundamental functionality of the software operates as intended and fulfills the defined criteria. Testers concentrate on the program's inputs and anticipated outputs, disregarding the underlying code that produces those outcomes.

Utilizing black box testing for the function mapping.c, we sought to confirm that its exterior behavior, as seen by end-users, corresponds with the anticipated functionality detailed in the milestone specification. This method enables us to evaluate if the function achieves its intended objective without exploring the complexities of its internal workings.

Conversely, white box testing, when applied to the function truck.c, utilizes an understanding of the core code and logic. It allows for a comprehensive analysis and verification of the function's core mechanisms, ensuring the code operates accurately and effectively. This testing phase is particularly beneficial for unique data structures and freshly developed functions, as it aids in detecting potential errors, edge cases, and performance enhancements through meticulous code analysis.

1. Explain the purpose of the automation hook for GIT and explain how it can improve the quality of the software in the project.

There is a script or tool called the automation hook for GIT that makes some tasks in the Git workflow automatic. For instance, it can run test cases automatically before letting you commit or push to the repository. Before being accepted, this hook makes sure that every change to the codebase passes a quality check that has already been set up. For example, unit tests or static code analysis can be used to do this. This makes the software better in several ways: Avoiding Mistakes: It keeps broken or untested code from getting into the shared repository, which lowers the risk of integration problems. Consistency: It makes sure that everyone on the team follows the same rules for coding and running tests, which makes the development process more consistent. Efficiency: Developers get feedback on their changes right away, which lets them fix problems quickly before they spread.